Tuesday, May 11, 2004 Version 1.0

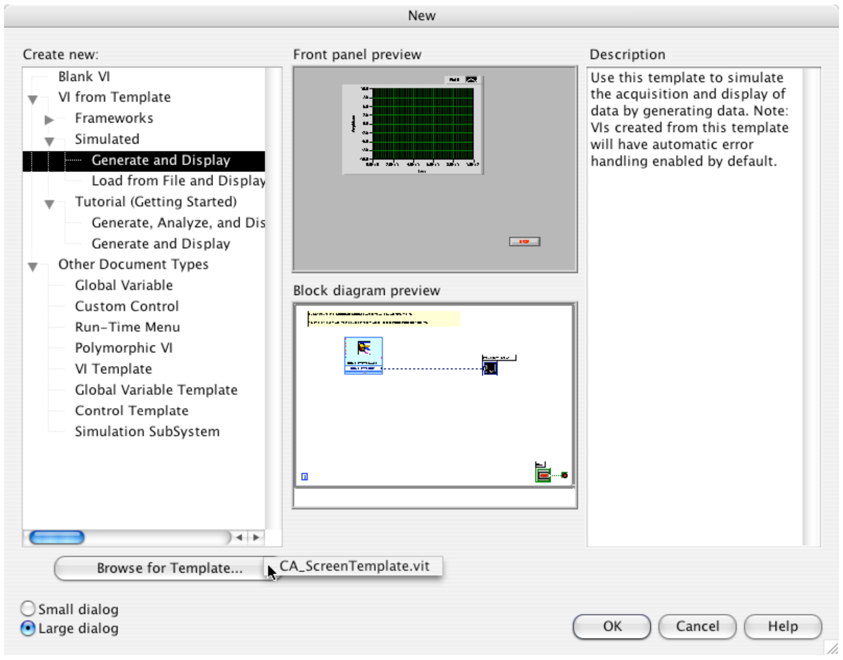
LabVIEW EPICS Console Screens

By Willem Blokland SNS

This program demonstrates the use of LabVIEW as an EPICS console screen. The setup is very similar to EDM (EPICS Display Manager) in that the setup process consists of selecting display elements, positioning the elements, and associating these with a PV (Process Variable). There is no programming involved in this stage. However, one can also add analysis of the data to the console screen using the standard LabVIEW graphical programming environment.

Creating a screen

To create a screen, the user starts by creating a new VI (select “new…” from the file menu) using the template CA_ScreenTemplate.vit, see Figure 1. This VI has the different front-panel controls or indicators that are supported for this demo.



Use this template to simulate the acquisition and display of data by generating data. Note: VIs created from this template will have automatic error handling enabled by default.

Figure 1. Creating a new console screen from a Template VI.

The demo supports waveform and chart graphs (single and multiple), tables, and single variable indicators and controls. To associate a PV with the indicator or control, you must enter the name(s) of the PV in the name of the indicator or control, see Figure 2.

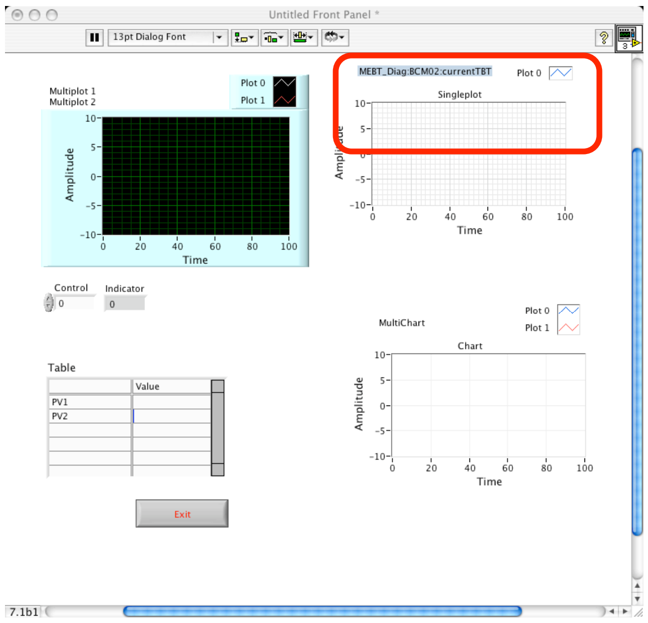


Figure 2. Associating a PV with a display.

Delete the indicator or controls you do not want and setup the plot you do want. LabVIEW support the naming of the axis, selection tools, auto-scaling, cursors, and more. Leave the EXIT button on the screen; this button is used to terminate the task that will acquire the EPICS data and update the indicators and read out the controls, see Figure 3.

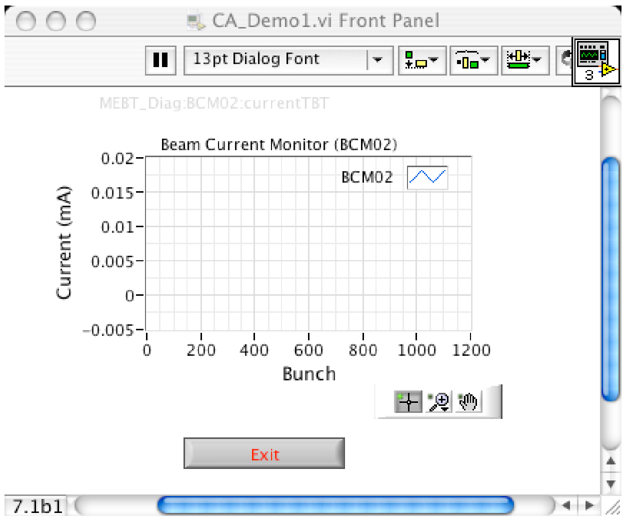
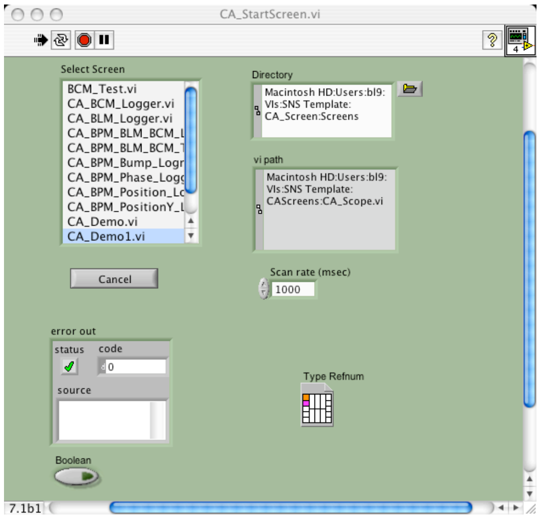


Figure 3. Editing and saving the template to CA_Demo1.vi

Save and close the VI in the directory for the screens. This can be anywhere you want it to be as long as you set it the same in the start screen utility, see below.

Running the screen

Open the utility CA_StartScreen.vi, set the directory to where you saved your screen, and run it. The utility will display all VIs in the folder. Select your newly created VI from the Select Screen lists by double clicking on its entry, see Figure 4.



Select Screen
BCM_Test.vi
CA_BCM_Logger.vi
CA_BLM_Logger.vi
CA_BPM_BLM_BCM_L
CA_BPM_BLM_BCM_T
CA_BPM_Bump_Logr
CA_BPM_Phase_Logg
CA_BPM_Position_Lo
CA_BPM_PositionY_L
CA_Demo.vi
CA_Demo1.vi

Directory: Macintosh HD:Users:bl9: VIs:SNS Template: CA_Screen:Screens

vi path: Macintosh HD:Users:bl9: VIs:SNS Template: CAScreens:CA_Scope.vi

Scan rate (msec): 1000

error out
status: ✓
code: 0
source:

Figure 4 Screen Startup utility.

The startup utility collects data on the front-panel indicators and controls. Indicators are considered as “in” PVs and controls as “out” PVs. The names of PVs are collected and the program checks if these exist. All EPICS Channel Access calls are made through the CA Client by A. Liyu. Next, a generic LabVIEW task starts in the background and opens channels to all the PVs. This task periodically scans the PVs and sends the data to the indicators of the screen VI Ca_Demo1.vi using references to the indicators. The screen with updated values is seen in Figure 5. Technically, the console screen VI does not have to be in a run state but the graph’s controls work better when the VI is in control mode. Controls on the front-panels are grouped in an event. This event is sent when the value of any control changes. The generic task catches this event and then sets the value of the PV. When the user pushes the Exit button, the generic task will terminate. The user will not see the task popup but can force it visible by selecting in the “Show VI Hierarchy” from the “Browse” menu.

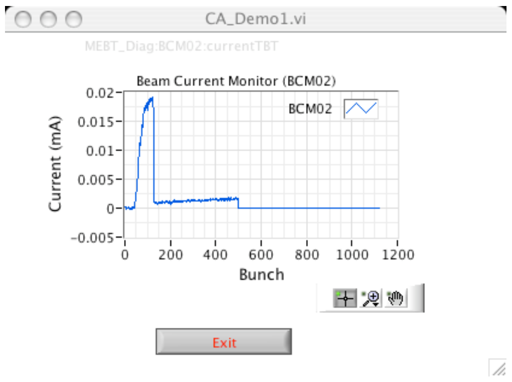


Figure 5. The demo console screen with data.

To support the run state of the console screen VIs, the screen template implements an event structure inside a while loop, see Figure 6.

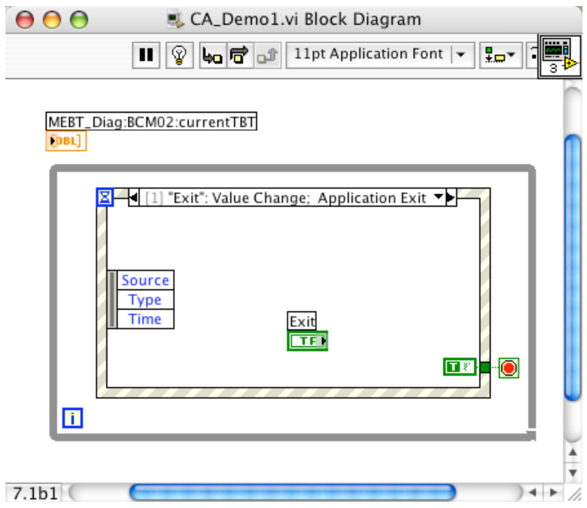

CA_Demo1.vi Block Diagram
11pt Application Font
MEBT_Diag:BCM02:currentTBT
DBL
[1] "Exit": Value Change; Application Exit
Source
Type
Time
Exit
TF
7.1b1

Figure 6. The diagram of the screen.

Performance

At this point the code is not optimized and the variant data type is used to pass data of different types to the indicators. I suspect this is not the most efficient method. Over a 100Mb/s Ethernet connection, a 1.25 Ghz Powerbook G4 can display 3 traces of 100 double points at 10Hz using a 10% CPU load.

Summary

The Demo shows how LabVIEW can be used as a console environment. The user doesn't have to program in LabVIEW but instead merely drop and drags indicators and controls and edits the names to create a working console screen. The current setup uses a screen startup utility but this VI can be placed within the screen VI to make it self-starting. As updates come in, the event structure of the screen VI can capture the updated data and a programmer can easily add data-analysis to the console screen VI. The demo uses the EPICS Channel Access protocol to obtain data but the same technique can be applied to other protocols as well, e.g. TCPORT or Acnet. While now the name of the front-panel element is used (except for the table indicator) one could also use the caption or even a text file (or dedicated VI tool) to create the associations between indicator and PV.